How To Edit a Dialog’s Enhanced Source Code

What Is The Enhanced Source Code Format ?

The enhanced source code format enables you to edit source code that has been generated by the Dialog Editor. You edit enhanced source code in a program editor window. When you edit a dialog, the Dialog Editor stores the results in internal structures. From these structures, source code is generated when you save, stow, list or execute any other system command on the dialog. Code is also generated when you refresh the program editor’s source code window.

You can edit enhanced source code as you do any other Natural user code. The source code syntax is subject to a number of formal conventions, however. For a documentation of the enhanced source code syntax, see Enhanced Source Code Format in the Dialog Component Reference documentation.

When you execute a system command on a dialog you have just edited in the program editor source code window, the Dialog Editor updates its internal structures and refreshes the source code window.

Note: The Dialog Editor preserves code layout only in the user code sections, such as event handlers.

The Dialog Editor supports the following source formats:

- 213. This is the format generated by Natural Version 2.1.3 (New Dimension). It is supported for input only. You cannot generate 2.1.3 format with Natural Version 3.1 and Version 3.2.
- 22C. This is the format generated by Natural Version 2.2.2. In Natural for Windows and Unix Version 4.1, dialogs can no longer be generated in this format. It, too, is supported for input only.
- 22D. This is the "enhanced" source-code format that from now on is the standard. It is generated for compiling, storing, and editing dialogs in Natural Version 2.2.3 and above.

The characteristics of the enhanced source code format are:

- Dialog sources are readable and printable without requiring conversion.
- Dialog sources consist only of legal and fully documented Natural syntax.
- Dialog sources can be edited textually using Program Editor functions such as scanning for and replacing text.
- Dialog sources can be displayed in the Natural Debugger.
- Dialog sources are larger than 213 or 22C format sources (by a factor between 1.25 and 3.5).
- Any code that can be generated with the Dialog Editor can also be coded manually. For example, if you "draw" a push-button control onto the user interface, the corresponding code is generated implicitly. You can also create this push-button control explicitly with the help of a source-code window that provides you with the functions of the Program Editor.
- You can switch between the Dialog Editor and the Program Editor by selecting the source code window or the dialog window. If you edit in either window, you need to synchronize your updates: (graphically) modifying the dialog locks the source code window and you may not make changes there. Correspondingly, if you change the source code, you may not make changes in the dialog window, which is locked. If your editor is locked, its status bar displays "Locked".

For dialogs in the old formats, this means:

- They remain unchanged until they are processed in the Dialog Editor. They can be compiled and executed in their old format.
- When you load them into the Dialog Editor, the dialogs are saved in the new format. If they are saved in the enhanced format, you must include the local data area NGULKEY1. Note that the storage size increases when the dialogs are saved.
- When you list or print them and you enable the "enhanced list mode" option, the dialogs are displayed using the enhanced source code format.

Avoiding Incompatibilities Between Dialog Editor And Program Editor

When you edit the enhanced source code format, note that some of the syntax elements accepted by the Program Editor are not accepted by the Dialog Editor. Enhanced source code editing is not intended as a new programming technique in addition to using the Dialog Editor:

- It may be syntactically acceptable to replace a dialog element’s numeric coordinate (a RECTANGLE-X attribute value) with a variable reference. The Dialog Editor, however, will not accept this when the changes are synchronized, and will prompt you when you issue a command requiring the source code.
- The Dialog Editor may accept a reference to a variable’s STRING attribute even if the variable is not declared, but the compiler will not accept this.

In the sections that are not user code, you should avoid such incompatibilities by adding only code that is acceptable to both the compiler and the Dialog Editor.

In the user code sections, such as in event-handler sections and in external or internal subroutines, your choice of programming techniques is not restricted by the Dialog Editor. In these sections, however, you have no visual editing support.

As a general rule, a mixed approach is often the best, especially when you use dialog-editor- generated code as a starting point.

Note: In the Dialog Editor, you can copy dialog elements to the clipboard and when you paste them into user code, they appear as text.

How To Use The Enhanced Source Code Format

To edit a dialog in the enhanced source code format

1. Load the dialog into the Dialog Editor.
2. From the "Dialog" menu, choose "Source Code".
   Or choose the "Source Code" toolbar button.
   Or press CTRL+ALT+C.

The dialog’s source code window appears and the program editor is loaded. This editor enables you to scan for text strings, replace them, and so on. For more information on how to use the program editor, refer to the Program Editor.

The enhanced source code format’s syntactical conventions are documented in the chapter Enhanced Source Code Format in the Dialog Component Reference documentation.

Enhanced source code can be listed and printed as usual. You can also scan for strings by using the Find option of the Edit menu.

Note: If you are replacing strings with this option, this can make a dialog source incompatible with the Dialog Editor.

Back to Event-Driven Programming Techniques.

